EECS 183 Fall 2021

Exam 2 with Solutions

Closed Book 3" x 5" notecard No Electronic Devices Closed Neighbor

All cell phones and other electronic devices, including smart watches, must be turned off and placed in your backpack

Instructions

1. After we indicate you may start, print your uniqname at the top of every page of this booklet.
2. After we indicate you may start the exam, verify that you have received every page of the exam. There are 16 pages in this booklet, including this cover page.
3. You may have and reference one 3" x 5" notecard.
4. Assume all code and code fragments are syntactically valid, unless otherwise specified.
5. Assume/use only standard C++11.
6. In all the given code, if any character looks like a space, it is one.
7. Write clearly. If we cannot read your writing, it will be marked wrong. This includes indentation and curly braces.
8. Sign below and print your name and UMID. 10 pts off if we cannot read your name or UMID.

_____

Clearly and legibly print your **UMID** above

"I have neither given nor received aid on this examination, nor have I concealed any violations of the Honor Code."

_____

SIGN your name above

_____

Clearly and legibly print your **full name (NOT uniqname)** above

Problem Number	Points Possible
Multiple Choice	90
Free Response 1	18
Free Response 2	18
Free Response 3	18
Free Response 4	27
Free Response 5	9
	180

You can use the area below for working out problems.

Multiple Choice [5 points each]

Record your choices for all multiple choice questions using the circles next to each question.

Fill the single circle corresponding to your answer completely, like this: Ⓐ ● Ⓒ Ⓓ Ⓔ

1. Which of the following about C++ classes is **false**?	Ⓐ Ⓑ Ⓒ Ⓓ Ⓔ

A. Classes hide low-level internal details through abstraction.
B. Classes include both data and functionality.
C. Class member functions can always be accessed using the dot operator given an instance of the class.
D. Class data members can be other class data types.
E. Class data members can be of different data types.

C

2. What will the state of the stream `inFile` be after executing the following code?
 	
`ifstream inFile;`
`inFile.open("triangles.txt");`
	Ⓐ Ⓑ Ⓒ

A. `inFile` will be in a good state, i.e., `inFile.good()` will be `true`
B. `inFile` will be in a fail state, i.e., `inFile.fail()` will be `true`
C. There is not enough information to know the status of stream states.

C

3. If the filestream `readFile` is in the fail state, which statement can restore the stream to a good state?	Ⓐ Ⓑ Ⓒ Ⓓ Ⓔ

A. `readFile.good();`
B. `readFile.clear();`
C. `!readFile.fail();`
D. `readFile.reset();`
E. `readFile.eof();`

B

4. Which of the expressions below will read and sum the integers from a file until the value of `0` is read or the end of the file is reached?

```
ifstream numFile;
numFile.open("integers.txt");
int val = 0;
int sum = 0;

while (/* what goes here? */) {
    sum += val;
}
```

Ⓐ Ⓑ Ⓒ Ⓓ

A. `val != 0 || numFile >> val`
B. `val != 0`
C. `numFile >> val || val != 0`
D. `numFile >> val && val != 0`

D

5. Which of the following about C++ arrays is **true**?

Ⓐ Ⓑ Ⓒ Ⓓ Ⓔ

A. The first valid index of an array is `1`
B. The number of elements of an array can be changed after declaration.
C. Each element of an array is of the data type `array`
D. To access elements of a 2-D array, nested loops must be used.
E. None of A, B, C, or D are **true**.

E

Consider the following function, which will be **used in the next two questions**.

```
/**
 * Requires: Nothing.
 * Modifies: count
 * Effects:  Sets count to the number of occurrences
 *           of letter in sentence
 */
void letterCount(string& sentence, char letter, int& count);
```

6. Given the following declarations:

```
string phrase = "Life is just for a while.";
char search = 'a';
int frequency = -1;
```

How many of the following are valid calls to the function `letterCount`?

```
letterCount(phrase, search, 10);
letterCount(phrase, 'a', frequency);
letterCount(phrase, search, frequency);
letterCount("", search, frequency);
letterCount("Life is just for a while.", 'i',
            frequency);
```

Ⓐ Ⓑ Ⓒ Ⓓ Ⓔ

A. None are valid calls.
B. 1
C. 2
D. 3
E. All are valid calls.

C

7. Could the parameter `sentence` be changed to pass-by-value, e.g.., change to `string sentence` (no &), and still fulfill the Effects clause of the RME?

Ⓐ Ⓑ Ⓒ

A. Yes.
B. No.
C. Without seeing the definition there is not enough information to know.

A

Consider the following code fragment, which will be **used in the next two questions**.

```
/**
 * Requires: size <= MAX_SIZE
 * Modifies: ??? Q8 what goes here ???
 * Effects:  ??? Q9 what goes here ???
*/
bool yeetArray(const int arr[MAX_SIZE], int size) {
    bool yeet = true;
    for (int i = size - 1; i >= 0; i--) {
        if (arr[i] != arr[size - i - 1]) {
            yeet = false;
        }
    }
    return yeet;
}
```

8. What is the Modifies clause of the RME for the function?	Ⓐ Ⓑ Ⓒ Ⓓ Ⓔ

A. `Modifies: Nothing.`
A
B. `Modifies: arr`
C. `Modifies: arr, size`
D. `Modifies: yeet`
E. `Modifies: i`

9. What is the Effects clause of the RME for the function?	Ⓐ Ⓑ Ⓒ Ⓓ Ⓔ

A. Effects: returns false when any elements of arr are different,
returns true otherwise

B. Effects: returns false when all elements of arr are different,
returns true otherwise

E

C. Effects: returns true when all elements of arr are sorted,
returns false otherwise

D. Effects: returns true when pairs of elements of arr are identical,
returns false otherwise

E. Effects: returns true when elements of arr are the same backwards
as forward, i.e., a palindrome, - example: racecar
returns false otherwise

The supplemental page separate from this exam, with the `Sheet` class definition, is relevant to the remaining questions on the exam.

10. Which of the following code snippets, in `main`, would create an instance of `Sheet`?	Ⓐ Ⓑ Ⓒ Ⓓ

A. `Sheet;`
B. `Sheet newSheet;`
C. `newSheet.Sheet();`
D. `Sheet newSheet(MAX_SIZE);`

B

11. Consider the following variable declaration, found in `main()`: `Sheet testSheet = Sheet();` Which of the following is a valid call to the `isValidIndex` function?	Ⓐ Ⓑ Ⓒ Ⓓ Ⓔ

A. `bool isValid = testSheet.isValidIndex();`
B. `bool isValid = testSheet.isValidIndex(1);`
C. `testSheet.isValidIndex(2);`
D. `isValidIndex(2);`
E. None of A, B, C, or D are valid calls to `isValidIndex`

E

12. Which of the following creates an array of `Sheet` instances named `multiSheet`?	Ⓐ Ⓑ Ⓒ Ⓓ Ⓔ

A. `Sheet[10];`
B. `multiSheet[10];`
C. `Sheet multiSheet;`
D. `Sheet multiSheet[10];`
E. `Sheet multiSheet = Sheet(10);`

D

13. Consider the array declaration of type `Sheet` from the previous problem. How would you print the size of the ***second element*** of the array `multiSheet`?	Ⓐ Ⓑ Ⓒ Ⓓ Ⓔ

B

A. `cout << multiSheet[1].size;`
B. `cout << multiSheet[1].getSize();`
C. `cout << multiSheet[2].getSize();`
D. `cout << multiSheet.getSize(1);`
E. `cout << multiSheet.getSize(2);`

14. Which of the following is a valid way to call the member function `reSize` given the following declarations?
 	
`Sheet tinySheet = Sheet();`
`int value = -1;`
	Ⓐ Ⓑ Ⓒ Ⓓ Ⓔ

D

A. `Sheet.reSize(tinySheet, value);`
B. `tinySheet.reSize();`
C. `value.reSize(tinySheet);`
D. `tinySheet.reSize(value);`
E. None of A, B, C, or D are valid calls to `reSize`

15. For an instance of the `Sheet` data type declared in the `main` function of a program, which class members can be accessed using the dot `.` operator?	Ⓐ Ⓑ Ⓒ Ⓓ

A

A. `public` class members
B. `private` class members
C. Both `public` and `private` class members
D. Neither `public` nor `private` class members

16. Which class members of the `Sheet` data type can be accessed *in the scope of a member function* of the `Sheet` class?	Ⓐ Ⓑ Ⓒ Ⓓ

C

A. `public` class members
B. `private` class members
C. Both `public` and `private` class members
D. Neither `public` or `private` class members

17. Which of the following is a correct way to implement the member function `getSize` just as it would appear in the file Sheet.cpp?	Ⓐ Ⓑ Ⓒ Ⓓ Ⓔ

A.
```
Sheet::getSize() {
    return size;
}
```

B.
```
Sheet::int getSize() {
    return size;
}
```

D

C.
```
int Sheet::getSize();
```

D.
```
int Sheet::getSize() {
    return size;
}
```

E.
```
int Sheet::getSize() {
    return getSize;
}
```

18. Which of the following about the `Sheet` class member `setGridValue` is **true**?	Ⓐ Ⓑ Ⓒ Ⓓ

D

A. `setGridValue` is a constructor function
B. `setGridValue` is a private member function
C. `setGridValue` will *always* modify `grid`
D. `setGridValue` can be called with an empty string `""` as an argument

Free Response 1 [18 points]

Implement the following function, which counts the number of elements in an array that are equal to a value provided as an argument to the function.

```
/**
 * Requires: size > 0 and size <= MAX_SIZE
 *           size is the number of rows and the number of columns for board
 * Modifies: numFound
 * Effects:  Sets numFound to the number of elements
 *           in board that match searchFor
 *           within the bounds of size number of rows and columns
 */
void countElements(int board[MAX_SIZE][MAX_SIZE], int searchFor,
                   int& numFound, int size) {
```

```
   // example solution

    numFound = 0;
    for (int row = 0; row < size; row++) {
        for (int col = 0; col < size; col++) {
            if (board[row][col] == searchFor) {
                numFound++;
            }
        }
    }
    return;
```

```
}
```

Free Response 2 [18 points]

Implement the following function, which reads the contents of a file and stores them in a string variable.

```
/**
 * Requires: ins stream in good state and a file has already
 *           been successfully opened
 * Modifies: ins, fileData
 * Effects:  Reads all words or phrases in a file,
 *           i.e., any alphanumeric blocks separated by whitespace,
 *           and assigns them to the string fileData.
 *           Adds a blank space, i.e., ' ',
 *           after each word or block of text
 *
 * Notes:    You may NOT use getline() in your solution
 *
 * Example 1: If a file contains:
 *            The Grasshopper Lies Heavy.
 *            Then fileData will be the string:
 *            "The Grasshopper Lies Heavy. "
 *
 * Example 2: If a file contains:
 *                1 Everything in life is
 *            just for a     while
 *
 *            Then fileData will be the string:
 *            "1 Everything in life is just for a while "
 */
void read(ifstream& ins, string& fileData);
```

*****Write your solution in the box *on the next page******

```
void read(ifstream& ins, string& fileData) {
```

```
    // example solution
    fileData = "";
    string word = "";
    while (ins >> word) {
        fileData += word;
        fileData += " ";
    }

    return;
```

```
}
```

The supplemental page separate from this exam, with the `Sheet` class definition, is relevant to the remaining questions on the exam.

Free Response 3 [18 points]

Implement the `Sheet` class member function **`reSize`** below, **just as it would appear in the file `Sheet.cpp`.** For your convenience, the function declaration from the `Sheet` class definition is repeated below.

```
/**
 * Requires: Nothing.
 * Modifies: size
 * Effects:  Adds adjustSize to size.
 *           If size + adjustSize is < 0 or > MAX_SIZE:
 *               sets size to closest of 0 and MAX_SIZE
 *
 * NOTE: Does *not* modify grid.
*/
void reSize(int adjustSize);
```

Be certain to include the function header (with function name, etc.) in your definition!

```
// example solution
void Sheet::reSize(int adjustSize) {
    size = size + adjustSize;

    /*  must check for invalid size
        can not use isValidIndex since that
        compares to size not MAX_SIZE
    */
    if (size < 0) {
        size = 0;
    }
    else if (size > MAX_SIZE) {
        size = MAX_SIZE;
    }
    return;

}
```

Free Response 4 [27 points]

Implement the `Sheet` class member function **`swapRows`** below, **just as it would appear in the file `Sheet.cpp`.** For your convenience, the function declaration from the `Sheet` class definition is repeated below.

```
/**
 * Requires: Nothing.
 * Modifies: grid
 * Effects:  If row1 and row2 both pass isValidIndex check:
 *              Swaps all elements in row1 with all elements of row2
 *              for columns within the bounds of [0, size)
 *           Otherwise, does nothing
*/
void swapRows(int row1, int row2);
```

Be certain to include the function header (with function name, etc.) in your definition!

```
// example solution
void Sheet::swapRows(int row1, int row2) {
    // do not go outside of size
    if (!isValidIndex(row1) || !isValidIndex(row2)) {
        return;
    }

    // swap each column between row1 and row2
    for (int col = 0; col < size; col++) {
        string temp = grid[row1][col];
        grid[row1][col] = grid[row2][col];
        grid[row2][col] = temp;
    }
    return;

}
```

Free Response 5 [9 points]

Now, complete the **`main`** function below to use the `Sheet` class. Write your solution statements below each comment.

```
int main() {
```

```
// declare a single Sheet instance named exam (this is done for you)
Sheet exam = Sheet();

// call reSize function using exam to add 10 rows and columns


exam.reSize(10);



// set the grid of exam at the top left corner to the value "hello"


exam.setGridValue(0, 0, "hello");



// swap the first(top) and second rows of exam


exam.swapRows(0, 1);
```

```
    return 0;
}
```